## LISTING OF CLAIMS

This listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Original) In a multiprocessing computer system having a plurality of processing devices, a method for identifying processes to be executed, the method comprising:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system.

2. (Original) The method of claim 1 comprising:

in the kernel responsible for selecting processes to execute amongst the plurality of processing devices:

receiving the identification of the second process as a process that is ready for execution in the multiprocessing computer system;

applying an affinity-based process selection technique to select the second process as a next process to execute on the first processing device;

executing the second process on the first processing device.

3. (Original) The method of claim 2 wherein the first processing device maintains at least a portion of the shared data accessed and released by the first process in a cache associated with the first processing device and wherein executing the second process on the first processing device comprises:

executing the second process to access the at least a portion of the shared data maintained in the cache associated with the first processing device, such that the first processing device does not have to access that at least a portion of the shared data from a main memory associated with the multiprocessing computer system.

4. (Original) The method of claim 2 wherein detecting when a first process executing on a first processing device releases access to shared data, attempting to identify a second process, and providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process are performed by a synchronization subsystem that operates independently of the kernel to control access to the share data.

5. (Original) The method of claim 4 wherein the synchronization subsystem operates in a Java Virtual Machine.

6. (Original) The method of claim 1 wherein attempting to identify a second process that formerly executed on the first processing device and that is awaiting to access shared data comprises:

reviewing execution state associated with respective blocked processes awaiting access to the shared data; and

if the execution state of a blocked process indicates that the blocked process formerly executed on the first processing device, identifying that blocked process as the second process.

7. (Original) The method of claim 6 wherein if multiple blocked processes include associated execution histories that indicate each form executed on the first processing device, identifying that blocked process as the second process comprises:

identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed least recently as compared to other blocked processes

awaiting access to the shared data.

8. (Original) The method of claim 6 wherein if multiple blocked processes include associated execution histories that indicate each form executed on the first processing device, identifying that blocked process as the second process comprises:

identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed most recently as compared to other blocked processes awaiting access to the shared data.

9. (Original) The method of claim 6 wherein:

if no blocked process has an associated execution state that indicates former execution on the first processing device, performing an alternate blocked process selection technique to identify the second process as a process that is ready for execution in the multiprocessing computer system.

10. (Original) The method of claim 9 wherein performing an alternate blocked process selection technique to identify the second process as a process that is ready for execution in the multiprocessing computer system comprises:

identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed least recently as compared to other blocked processes awaiting access to the shared data.

11. (Original) The method of claim 10 wherein identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed least recently as compared to other blocked processes awaiting access to the shared data comprises:

applying a forcible migration technique to the identified blocked process that has an execution state indicating that the blocked process executed formerly on a

processing device other than the first processing device to cause the identified blocked process to migrate to the first processing device when executed by the kernel and to be identified to the kernel as the second process for execution on the first processing device.

12. (Original) The method of claim 9 wherein performing an alternate blocked process selection technique to identify the second process as a process that is ready for execution in the multiprocessing computer system comprises:

applying a forcible migration technique to at least one blocked process that has an execution state indicating that the blocked process executed formerly on a processing device other than the first processing device and that is awaiting access to the shared data to cause the at least one blocked process to migrate to the first processing device when executed by the kernel and to be identified to the kernel as the second process for execution on the first processing device.

13. (Original) The method of claim 12 comprising:

in the kernel responsible for selecting processes to execute amongst the plurality of processing devices:

detecting that the forcible migration technique has been applied to the at least one blocked process that has an execution state indicating that the blocked process executed most recently on a processing device other than the first processing device and that is awaiting access to the share data and in response, migrating the at least one blocked process for execution on the first processing device; and

applying a non-affinity-based process selection technique to select the at least one blocked process as a next process to execute on the first processing device.

14. (Original) The method of claim 1 wherein:

the plurality of processing devices includes at least two processing devices that share a common cache with each other;

attempting to identify a second process that formerly executed on the first

processing device and that is awaiting access to the shared data cannot identify a second process that formerly executed on the first processing device; and

wherein the method comprises:

attempting to identify the second process as a process that:

i) formerly executed at least one process of the at least two processors that share a common cache; and

ii) is awaiting access to the shared data.

15. (Original) The method of claim 14 wherein the at least two processing device that share a common cache with each other comprises different core processors on a common processing die.

16. (Original) The method of claim 1 comprising:

detecting an execution behavior pattern for processes that access shared data;

determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, and if the execution behavior pattern statistically meets the threshold associated with patterns of access to the shared data, causing a synchronization subsystem responsible for controlling access to the shared data to perform the operations of:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system; and if the execution behavior pattern statistically does not meet the threshold associated with patterns of access to the share data, causing the kernel to perform a normal

successor selection process for selection of processes to succeed each other during execution.

17. (Original) The method of claim 16 wherein detecting an execution behavior pattern for processes that access the shared data and determining if the execution behavior pattern statistically meets a threshold are performed periodically such that the multiprocessing computer system selects processes to succeed execution of other processes according to at least one of:

the normal successor selection process performed by the kernel;

the affinity based wakeup locality successor selection process performed by a synchronization subsystem that controls access to the share data.

18. (Original) The method of claim 16 wherein detecting an execution behavior pattern for processes that access the shared data comprises:

calculating an amount of blocked acquire operations for access to the shared data that have occurred over a period of time, the blocked acquire operations performed by processes that attempt access to the shared data but that result in the process being blocked for access to the shared data by the synchronization scheduler; and

wherein determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data comprises:

determining if the amount of blocked acquire operations meets a threshold.

19. (Original) The method of claim 16 wherein detecting an execution behavior pattern for processes that access the shared data comprises:

calculating an average hold time for accesses to the shared data that have occurred over a period of time performed by processes that access to the shared data; and

wherein determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data comprises:

determining if the average hold time for accesses to the shared data meets a threshold.

20. (Original) In a multiprocessing computer system having a plurality of processing devices, a method for identifying processes to be executed, the method comprising:

detecting an execution behavior pattern for processes that access shared data;

determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, and if the execution behavior pattern statistically meets the threshold associated with patterns of access to the shared data, causing a synchronization subsystem responsible for controlling access to the shared data to perform an affinity-based wakeup locality successor selection process comprising:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system; and if the execution behavior pattern statistically does not meet the threshold associated with patterns of access to the share data, causing the kernel to perform a normal successor selection process for selection of processes to succeed each other during execution.

21. (Original) The method of claims 20 wherein detecting an execution behavior pattern for processes that access the shared data and determining if the execution behavior pattern statistically meets a threshold are performed periodically such that the multiprocessing computer system selects processes to succeed execution of other

processes according to at least one of:

the normal successor selection process performed by the kernel;

the affinity based wakeup locality successor selection process performed by a synchronization subsystem that controls access to the shared data.

22. (Original) The method of claim 20 wherein detecting an execution behavior pattern for processes that access the shared data comprises:

calculating an amount of blocked acquire operations for access to the shared data that have occurred over a period of time, the blocked acquire operations performed by processes that attempt access to the shared data but that result in the process being blocked for access to the shared data by the synchronization scheduler; and

wherein determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data comprises:

determining if the amount of blocked acquire operations meets a threshold.

23. (Original) The method of claim 20 wherein detecting an execution behavior pattern for processes that access the shared data comprises:

calculating an average hold time for accesses to the shared data that have occurred over a period of time performed by processes that access to the shared data; and

wherein determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data comprises:

determining if the average hold time for accesses to the shared data meets a threshold.

24. (Original) A multiprocessing computer system comprising:

a plurality of processing devices each having at least one associated cache;

a memory system;

an interconnection mechanism coupling the memory system and the plurality of

processing devices;

wherein the memory system is further encoded with a synchronization subsystem that operates to identify processes to be executed by performing the operations of:

detecting when a first process executing on a first processing device of the plurality of processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to the kernel operating in the multiprocessing computer system responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system.

25. (Original) The multiprocessing computer system of claim 24 wherein the kernel responsible for selecting processes to execute amongst the plurality of processing devices performs the operations of:

receiving the identification of the second process as a process that is ready for execution in the multiprocessing computer system;

applying an affinity-based process selection technique to select the second process as a next process to execute on the first processing device;

executing the second process on the first processing device.

26. (Original) The multiprocessing computer system of claim 25 wherein the first processing device maintains at least a portion of the shared data accessed and released by the first process in a cache associated with the first processing device and wherein when the kernel executes the second process on the first processing device the kernel performs the operations of:

executing the second process to access the at least a portion of the shared

data maintained in the cache associated with the first processing device, such that the first processing device does not have to access that at least a portion of the shared data from a main memory associated with the multiprocessing computer system.

27. (Original) The multiprocessing computer system of claim 25 wherein detecting when a first process executing on a first processing device releases access to shared data, attempting to identify a second process, and providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process are performed by the synchronization subsystem that operates independently of the kernel to control access to the share data.

28. (Original) The multiprocessing computer system of claim 27 wherein the synchronization subsystem operates in a Java Virtual Machine encoded within the memory and that operates in the multiprocessing computer system.

29. (Original) The multiprocessing computer system of claim 24 wherein when the synchronization subsystem performs the operation of attempting to identify a second process that formerly executed on the first processing device and that is awaiting to access shared data, the synchronization subsystem performs the operation of:

reviewing execution state associated with respective blocked processes awaiting access to the shared data; and

if the execution state of a blocked process indicates that the blocked process formerly executed on the first processing device, identifying that blocked process as the second process.

30. (Original) The multiprocessing computer system of claim 29 wherein if multiple blocked processes include associated execution histories that indicate each form executed on the first processing device, the synchronization subsystem performs the operation of identifying that blocked process as the second process by performing the operations of:

identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed least recently as compared to other blocked processes awaiting access to the shared data.

31. (Original) The multiprocessing computer system of claim 29 wherein if multiple blocked processes include associated execution histories that indicate each form executed on the first processing device, the synchronization subsystem performs the operation of identifying that blocked process as the second process by performing the operations of:

identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed most recently as compared to other blocked processes awaiting access to the shared data.

32. (Original) The multiprocessing computer system of claim 29 wherein:

if no blocked process has an associated execution state that indicates former execution on the first processing device, the synchronization subsystem performs the operation of performing an alternate blocked process selection technique to identify the second process as a process that is ready for execution in the multiprocessing computer system.

33. (Original) The multiprocessing computer system of claim 32 wherein when the synchronization subsystem performs the operation of performing an alternate blocked process selection technique to identify the second process as a process that is ready for execution in the multiprocessing computer system, the synchronization subsystem performs the operation of:

identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed least recently as compared to other blocked processes

awaiting access to the shared data.

34. (Original) The multiprocessing computer system of claim 33 wherein when the synchronization subsystem performs the operation of identifying, as the second process that is ready for execution in the multiprocessing computer system, a blocked process that is awaiting access to the shared data and that executed least recently as compared to other blocked processes awaiting access to the shared data, the synchronization subsystem performs the operation of:

applying a forcible migration technique to the identified blocked process that has an execution state indicating that the blocked process executed formerly on a processing device other than the first processing device to cause the identified blocked process to migrate to the first processing device when executed by the kernel and to be identified to the kernel as the second process for execution on the first processing device.

35. (Original) The multiprocessing computer system of claim 32 wherein when the synchronization subsystem performs the operation of performing an alternate blocked process selection technique to identify the second process as a process that is ready for execution in the multiprocessing computer system, the synchronization subsystem performs the operation of:

applying a forcible migration technique to at least one blocked process that has an execution state indicating that the blocked process executed formerly on a processing device other than the first processing device and that is awaiting access to the shared data to cause the at least one blocked process to migrate to the first processing device when executed by the kernel and to be identified to the kernel as the second process for execution on the first processing device.

36. (Original) The multiprocessing computer system of claim 35 wherein the kernel responsible for selecting processes to execute amongst the plurality of processing devices performs the operations of:

detecting that the forcible migration technique has been applied to the at least one blocked process that has an execution state indicating that the blocked process executed most recently on a processing device other than the first processing device and that is awaiting access to the share data and in response, migrating the at least one blocked process for execution on the first processing device; and

applying a non-affinity-based process selection technique to select the at least one blocked process as a next process to execute on the first processing device.

37. (Original) The multiprocessing computer system of claim 24 wherein:

the plurality of processing devices includes at least two processing devices that share a common cache with each other;

the synchronization subsystem performs the operation of attempting to identify a second process that formerly executed on the first processing device and that is awaiting access to the shared data cannot identify a second process that formerly executed on the first processing device; and

wherein the the synchronization subsystem performs the operation of attempting to identify the second process as a process that:

i) formerly executed at least one process of the at least two processors that share a common cache; and

ii) is awaiting access to the shared data.

38. (Original) The multiprocessing computer system of claim 37 wherein the at least two processing device that share a common cache with each other comprises different core processors on a common processing die.

39. (Original) The multiprocessing computer system of claim 24 wherein the synchronization subsystem performs the operation of:

detecting an execution behavior pattern for processes that access shared data; and

determining if the execution behavior pattern statistically meets a threshold

associated with patterns of access to the shared data, and if the execution behavior pattern statistically meets the threshold associated with patterns of access to the shared data, the synchronization subsystem responsible for controlling access to the shared data to perform the operations of:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system; and if the execution behavior pattern statistically does not meet the threshold associated with patterns of access to the share data, the synchronization subsystem causes the kernel to perform a normal successor selection process for selection of processes to succeed each other during execution.

40. (Original) The multiprocessing computer system of claim 39 wherein when the synchronization subsystem performs the operations of detecting an execution behavior pattern for processes that access the shared data and determining if the execution behavior pattern statistically meets a threshold, the synchronization subsystem performs these operations periodically such that the multiprocessing computer system selects processes to succeed execution of other processes according to at least one of:

the normal successor selection process performed by the kernel;

the affinity based wakeup locality successor selection process performed by a synchronization subsystem that controls access to the share data.

41. (Original) The multiprocessing computer system of claim 39 wherein when the

synchronization subsystem performs the operation of detecting an execution behavior pattern for processes that access the shared data, the synchronization subsystem performs the operations of:

calculating an amount of blocked acquire operations for access to the shared data that have occurred over a period of time, the blocked acquire operations performed by processes that attempt access to the shared data but that result in the process being blocked for access to the shared data by the synchronization scheduler; and

wherein when the synchronization subsystem performs the operation of determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, the synchronization subsystem performs the operation of:

determining if the amount of blocked acquire operations meets a threshold.

42. (Original) The multiprocessing computer system of claim 39 wherein when the synchronization subsystem performs the operation of detecting an execution behavior pattern for processes that access the shared data the synchronization subsystem performs the operation of:

calculating an average hold time for accesses to the shared data that have occurred over a period of time performed by processes that access to the shared data; and

wherein when the synchronization subsystem performs the operation of determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, the synchronization subsystem performs the operation of:

determining if the average hold time for accesses to the shared data meets a threshold.

43. (Original) A multiprocessing computer system comprising:

a plurality of processing devices each having at least one associated cache;

a memory system;

an interconnection mechanism coupling the memory system and the plurality of processing devices;

wherein the memory system is encoded with a synchronization subsystem that operates to identify processes to be executed by performing the operations of:

detecting an execution behavior pattern for processes that access shared data;

determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, and if the execution behavior pattern statistically meets the threshold associated with patterns of access to the shared data, the synchronization subsystem responsible for controlling access to the shared data performs an affinity-based wakeup locality successor selection process comprising the operations of:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system; and

if the execution behavior pattern statistically does not meet the threshold associated with patterns of access to the share data, the synchronization subsystem causes the kernel to perform a normal successor selection process for selection of processes to succeed each other during execution.

44. (Original) The multiprocessing computer system of claim 43 wherein detecting an execution behavior pattern for processes that access the shared data and determining if the execution behavior pattern statistically meets a threshold are performed periodically by the synchronization subsystem such that the multiprocessing computer

system selects processes to succeed execution of other processes according to at least one of:

the normal successor selection process performed by the kernel;

the affinity based wakeup locality successor selection process performed by a synchronization subsystem that controls access to the shared data.

45. (Original) The multiprocessing computer system of claim 43 wherein when the synchronization subsystem performs the operation of detecting an execution behavior pattern for processes that access the shared data the synchronization subsystem performs the operation of:

calculating an amount of blocked acquire operations for access to the shared data that have occurred over a period of time, the blocked acquire operations performed by processes that attempt access to the shared data but that result in the process being blocked for access to the shared data by the synchronization scheduler; and

wherein when the synchronization subsystem performs the operation of determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, the synchronization subsystem performs the operation of:

determining if the amount of blocked acquire operations meets a threshold.

46. (Original) The multiprocessing computer system of claim 43 wherein when the synchronization subsystem performs the operation of detecting an execution behavior pattern for processes that access the shared data, the synchronization subsystem performs the operation of:

calculating an average hold time for accesses to the shared data that have occurred over a period of time performed by processes that access to the shared data; and

wherein determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data comprises:

determining if the average hold time for accesses to the shared data meets a threshold.

47. (Original) A computer program product having a computer-readable medium including computer program logic encoded thereon that, when performed in a multiprocessing computer system identifies processes to be executed via the operations of:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system.

48. (Original) A computer program product having a computer-readable medium including computer program logic encoded thereon that, when performed in a multiprocessing computer system identifies processes to be executed via the operations of:

detecting an execution behavior pattern for processes that access shared data;

determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, and if the execution behavior pattern statistically meets the threshold associated with patterns of access to the shared data, causing a synchronization subsystem responsible for controlling access to the shared data to perform an affinity-based wakeup locality successor selection process comprising:

detecting when a first process executing on a first processing device releases access to shared data;

in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system; and

if the execution behavior pattern statistically does not meet the threshold associated with patterns of access to the share data, causing the kernel to perform a normal successor selection process for selection of processes to succeed each other during execution.

49. (Original) A multiprocessing computer system comprising:

a plurality of processing devices each having at least one associated cache;

a memory system;

an interconnection mechanism coupling the memory system and the plurality of processing devices;

wherein the memory system is encoded with a synchronization subsystem that operates to provide a means to identify processes to be executed, such means including:

means for detecting an execution behavior pattern for processes that access shared data;

means for determining if the execution behavior pattern statistically meets a threshold associated with patterns of access to the shared data, and if the execution behavior pattern statistically meets the threshold associated with patterns of access to the shared data, the synchronization subsystem responsible for controlling access to the shared data performs an affinity-based wakeup locality successor selection process that includes:

means for detecting when a first process executing on a first processing device releases access to shared data;

means for in response to the first process releasing access to the shared data, attempting to identify a second process that:

i) formerly executed on the first processing device; and

ii) is awaiting access to the shared data; and

means for providing, to a kernel responsible for selecting processes to execute amongst the plurality of processing devices, an identification of the second process as a process that is ready for execution in the multiprocessing computer system; and if the execution behavior pattern statistically does not meet the threshold associated with patterns of access to the share data, the synchronization subsystem includes a means for causing the kernel to perform a normal successor selection process for selection of processes to succeed each other during execution.